Design Document

Kameron Damaska Drew Borneman

December 6, 2016

Contents

JFreeChart Library

JFreeChart is a free Java library that produces charts. Our project utilizes this library to visualize the array as a bar graph at each iteration of a sorting algorithm.

Project Files

Sorting

Sort

`Sort.java` is the abstract class from all of our sorting algorithms are extended. The class includes a constructor that initializes the array that will be sorted.

This class includes the following methods:

- `sort`: An abstract method that must be overwritten by all classes that extend it. This sorts the array in the Sort object.
- `getArray`: Returns the array in the Sort object.
- `getSortedArray`: Checks to see if the array is already sorted. If it is, return the array. If not, the array is sorted and then returned.
- `isSorted`: Returns whether or not the array is already sorted.
- `updateGraph`: Updates the visualization of the array.

MergeSort

`MergeSort.java` is a class that implements the merge sort algorithm.

This class includes the following methods:

- `sort`: This method sorts the array in the MergeSort object using the merge sort algorithm.
- `merge`: Merges two sorted subarrays into one sorted subarray.

InsertionSort

`InsertionSort.java` is a class that implements the insertion sort algorithm.

The only method in this class is:

- `sort`, which sorts the array in the InsertionSort object using the insertion sort algorithm.

HeapSort

`HeapSort.java` is a class that implements the heap sort algorithm.

This class includes the following methods:

- `sort`: This method sorts the array in the HeapSort object using the heap sort algorithm.
- `swap`: Swaps two values within the array.
- `bubbleUp`: Moves an element in the heap to it's correct location using max heap logic.
- `buildHeap`: Builds a max heap.

QuickSort

`QuickSort.java` is a class that implements the quick sort algorithm.

This class includes the following methods:

- `sort`: Sorts the array in the QuickSort object using the quick sort algorithm.
- `partition`: This method uses the last value in a subarray, and puts all values less than it to the left and greater than it to the right.
- `swap`: Swaps two values within the array.

ShellSort

`ShellSort.java` is a clas that implements the shell sort algorithm.

This class includes the following methods:

- `sort`: Sorts the array in the ShellSort object using shell sort.

Visuals

GraphArray

`GraphArray.java`creates a frame that holds our graph.

This class includes the following methods:

- `graph`: Updates the graph for the new array.

Main Class

The main class in our project currently `VisualSorting.java`. The main method in this class takes two inputs: a String representing the sort name ("merge sort", "shell sort", etc), and an integer representing the array length. The proram then launches a window that graphs the step-by-step process of the sorting algorithm.

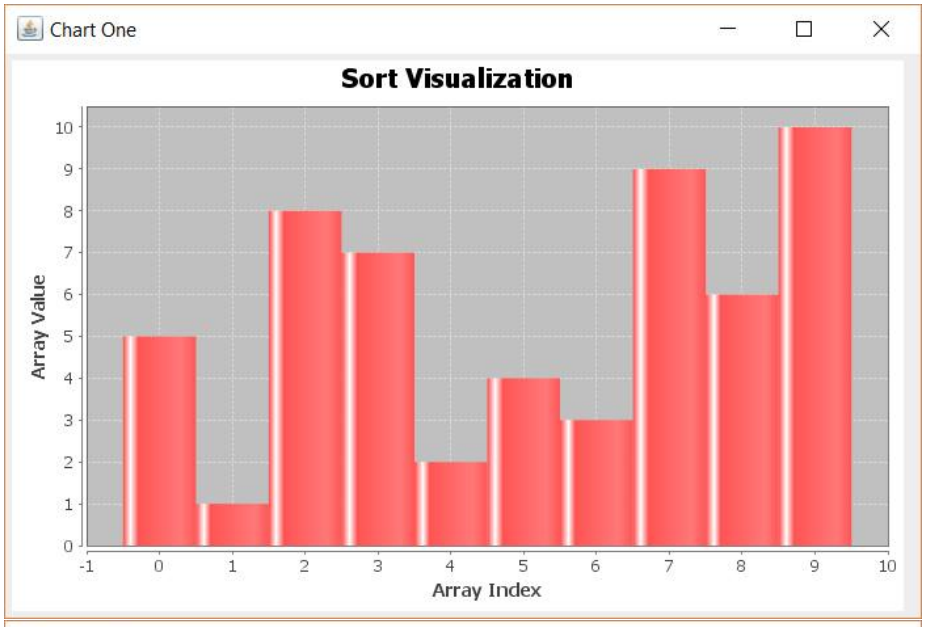
Methods

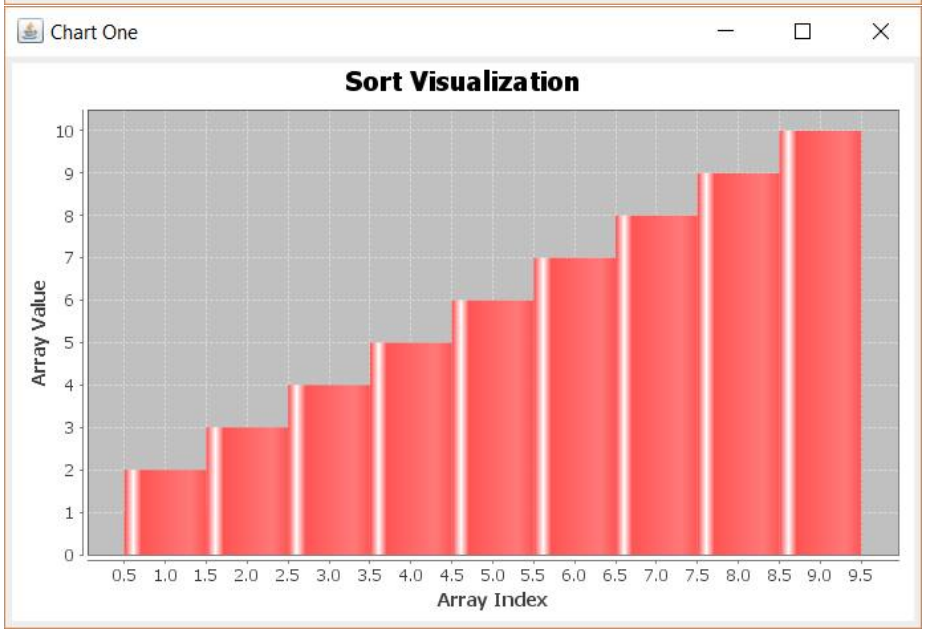
This class has the following methods:

- `launchProgram`: Launches the graph for a given sorting algorithm and array length.
- `parseString`: Removes all whitespace and decapitalizes the sorting algorithm name. This helps reduce input errors.
- `randomArray`: Generates a randomized array that will be sorted.
- `isValidSort`: Checks if the String input is a sorting algorithm in this program.

Output

MergeSort before and after with array size 10:





Moving Forward

Development from now until the final project submission will focus on developing a GUI to create a more user-friendly program. Additionally, we want to create the option to compare sorting algorithms, by opening multiple graphs in the frame. Finally, we will improve the look of the graph overall and do more extensive bug-testing.